IN THE SPECIFICATION:

The specification as amended below with replacement paragraphs shows added text with underlining and deleted text with ~~strikethrough~~.

Please REPLACE the paragraph beginning at page 9, line 22, with the following paragraph:

As described above, the character-recognition pre-processing apparatus 1 of the present invention employs a configuration such that when a string of characters, such as English characters, which produces a difference between ~~vitiation~~ variation in the envelope formed on the upper side of the character string and variation in the envelope formed on the lower side of the character string is subjected to character recognition, the orientation (upright or inverted) of the character string is detected through detection of the two variations, without use of character recognition. Thus, the orientation (upright or inverted) of a character string can be detected while minimizing the load imposed on a CPU.

Please REPLACE the paragraphs beginning at page 18, line 31, and ending on page 26, line 24, with the following paragraphs:

Specifically, in first step S21, the orientation judgment program 25 substitutes "0" for each of variables "Upright," "Inverted," and "Indefinite."

In subsequent step S22, the orientation judgment program 25 extracts character strings from a selected name-card image to be processed (a different name-card image is sequentially selected from the name-card images obtained by the inclination correction program 24). Specifically, by use of labeling processing, which is commonly used in image processing, the orientation judgment program 25 allots the same label to black pixels connected to one another (character portions) to thereby extract the character portions, and judges character portions which are not separated from one another by a predetermined distance to belong to the same character string and merges them, whereby character strings contained in the selected name-card image are extracted.

Specifically, in the case of a name-card image shown in FIG. 14, the orientation judgment program 25 extracts eight character strings, each of which is surrounded by a rectangle (smallest rectangle, which will be described later).

In subsequent step S23, the orientation judgment program 25 judges whether all the character strings extracted in step S22 have been selected. When having judged that not all the

character strings have been selected, the orientation judgment program 25 proceeds to step S24 and selects an unselected character string from the character strings extracted in step S22. In subsequent step S25, the orientation judgment program 25 sets smallest rectangles that surround the selected character strings, respectively, as shown in FIG. 14.

In subsequent step S26, the orientation judgment program 25 specifies the positions of the characters within the smallest rectangles. In the case of an example shown in FIG. 15a, the positions of characters "H," "o," "w," "a," "r," "d," "B," "r," "o," "w," and "n" are specified.

The above-described processing for specifying character positions are performed, making use of spaces present between characters. However, in some cases, no space is present between characters. In such a case, the orientation judgment program 25 regards the height of the smallest rectangle (the length in the direction perpendicular to the direction of arrangement of characters) as the height of characters, and estimates the width of each character on the basis of the estimated character width. Subsequently, the orientation judgment program 25 splits the character string into characters by use of the estimated width. Thus, the positions of the individual characters are specified or determined.

In subsequent step S27, the orientation judgment program 25 defines a plurality of (e.g., five) sampling lines at equal intervals at each of the character positions specified in step S26; detects the distance between the character region and the lower edge of the smallest rectangle as measured along each sampling line; and selects the shortest distance among the distances measured along the five sampling lines. In this manner, for each of the character positions specified in step S26, the orientation judgment program 25 calculates the shortest distance between the character region and the lower edge of the smallest rectangle. Further, the orientation judgment program 25 calculates a variance of the shortest distances.

Specifically, in the case of an example shown in FIG. 15b, the orientation judgment program 25 calculates the shortest distance X_H between the character region of "H" and the lower edge of the smallest rectangle, the shortest distance X_o between the character region of "o" and the lower edge of the smallest rectangle, the shortest distance X_w between the character region of "w" and the lower edge of the smallest rectangle, the shortest distance X_a between the character region of "a" and the lower edge of the smallest rectangle, the shortest distance X_r between the character region of "r" and the lower edge of the smallest rectangle, the shortest distance X_d between the character region of "d" and the lower edge of the smallest rectangle, the shortest distance X_B between the character region of "B" and the lower edge of the smallest rectangle, the shortest distance X_r between the character region of "r" and the lower edge of the

smallest rectangle, the shortest distance X_o between the character region of "o" and the lower edge of the smallest rectangle, the shortest distance X_w between the character region of "w" and the lower edge of the smallest rectangle, and the shortest distance X_n between the character region of "n" and the lower edge of the smallest rectangle. Subsequently, the orientation judgment program 25 calculates a variance of these shortest distances.

The thus-calculated variance assumes a small value when the shortest distances involve a small variation, and assumes a large value when the shortest distances involve a large variation.

In subsequent step S28, for each of the character positions specified in step S26, the orientation judgment program 25 calculates the distance between the character region and the upper edge of the smallest rectangle as measured along each sampling line set in step S27, and selects the shortest distance among them. In this manner, for each of the character positions specified in step S26, the orientation judgment program 25 calculates the shortest distance between the character region and the upper edge of the smallest rectangle. Further, the orientation judgment program 25 calculates a variance of the shortest distances.

Specifically, in the case of an example shown in FIG. 15b, the orientation judgment program 25 calculates the shortest distance X_H between the character region of "H" and the upper edge of the smallest rectangle, the shortest distance X_o between the character region of "o" and the upper edge of the smallest rectangle, the shortest distance X_w between the character region of "w" and the upper edge of the smallest rectangle, the shortest distance X_a between the character region of "a" and the upper edge of the smallest rectangle, the shortest distance X_r between the character region of "r" and the upper edge of the smallest rectangle, the shortest distance X_d between the character region of "d" and the upper edge of the smallest rectangle, the shortest distance X_B between the character region of "B" and the upper edge of the smallest rectangle, the shortest distance X_r between the character region of "r" and the upper edge of the smallest rectangle, the shortest distance X_o between the character region of "o" and the upper edge of the smallest rectangle, the shortest distance X_w between the character region of "w" and the upper edge of the smallest rectangle, and the shortest distance X_n between the character region of "n" and the upper edge of the smallest rectangle. Subsequently, the orientation judgment program 25 calculates a variance of these shortest distances.

The thus-calculated variance assumes a small value when the shortest distances involve a small variation, and assumes a large value when the shortest distances involve a large variation.

In subsequent step S29, the orientation judgment program 25 judges whether a significant difference is present between the variance calculated in step S27 and the variance calculated in step S28. When having judged that no significant difference is present between the variances, the orientation judgment program 25 proceeds to step S310. After incrementing the value of variable "Indefinite" by one in step S310, the orientation judgment program 25 returns to step S23 in order to process the next character string.

That is, in the case where English characters which constitute a character string are all capital letters, as shown in FIG. 16b, the variance calculated in step S27 (the variance of the shortest distances between the respective character regions and the lower edge of the smallest rectangle) becomes substantially equal to the variance calculated in step S28 (the variance of the shortest distances between the respective character regions and the upper edge of the smallest rectangle). In such a case, the value of variable "Indefinite" is incremented by one.

When having judged that a significant difference is present between the variance calculated in step S27 and the variance calculated in step S28, the orientation judgment program 25 proceeds to step ~~S11~~ S31 (the processing flow of FIG. 10). In step S311, the orientation judgment program 25 judges whether the variance calculated in step S28 is greater than the variance calculated in step S27. When having judged that the variance calculated in step S28 is greater than the variance calculated in step S27, the orientation judgment program 25 proceeds to step S312 in order to increment the value of variable "Upright" by one. Subsequently, the orientation judgment program 25 returns to step S23 in order to process the next character string.

That is, in the case where some of English characters which constitute a character string are lowercase letters, and the character string is in an upright state, the lower edge of the smallest rectangle serves as the base line of the characters. In such a case, as shown in FIG. 16a, 16c, and 16d, the variance calculated in step S28 (the variance of the shortest distances between the respective character regions and the upper edge of the smallest rectangle) becomes greater than the variance calculated in step S27 (the variance of the shortest distances between the respective character regions and the lower edge of the smallest rectangle). Therefore, in such a case, the value of variable "Upright" is incremented by one.

When having judged in step S131 that the variance calculated in step S28 is smaller than the variance calculated in step S27, the orientation judgment program 25 proceeds to step S313 in order to increment the value of variable "Inverted" by one. Subsequently, the orientation judgment program 25 returns to step S23 in order to process the next character string.

That is, in the case where a character string is in an inverted state (rotated 180 degrees), the upper edge of the smallest rectangle serves as the base line of the characters. In such a case, the variance calculated in step S28 (the variance of the shortest distances between the respective character regions and the upper edge of the smallest rectangle) becomes smaller than the variance calculated in step S27 (the variance of the shortest distances between the respective character regions and the lower edge of the smallest rectangle). Therefore, in such a case, the value of variable "Inverted" is incremented by one.

When, during the repeated execution of the processing in steps S23 to S313, the orientation judgment program 25 judges in steps S23 that the above-described processing has been completed for all the character strings contained in the name-card image, the orientation judgment program 25 proceeds to step S314. In step S314, the orientation judgment program 25 judges whether a significant difference is present between the count value of variable "Upright" and the count value of variable "Inverted." When having judged that a significant difference is present between the two count values, the orientation judgment program 25 proceeds to step S315. In step S135, the orientation judgment program 25 judges whether the count value of variable "Upright" is greater than the count value of variable "Inverted."

When having judged through the above-described judgment processing that the count value of variable "Upright" is greater than the count value of variable "Inverted," the orientation judgment program 25 judges that the name-card image selected to be subjected to processing is in an upright state (a state shown in FIG. 13a). Since the character string can undergo character recognition without rotation, the orientation judgment program 25 proceeds to step S136. In step S316, the orientation judgment program 25 performs processing for starting the character recognition program 28 in order to perform character recognition for the selected name-card image.

When having judged through the above-described judgment processing that the count value of variable "Upright" is less than the count value of variable "Inverted"; i.e., the count value of variable "Inverted" is greater than the count value of variable "Upright," the orientation judgment program 25 judges that the name-card image selected to be subjected to processing is in an inverted state (a state shown in FIG. 13b), and proceeds to step S317. After rotating the selected image by 180 degrees in order to bring the image into an upright state in step S317, the orientation judgment program 25 performs processing for starting the character recognition program 28 in order to perform character recognition for the rotated name-card image.

When having judged in step S134 that no significant difference is present between the

count value of variable "Upright" and the count value of variable "Inverted," the orientation judgment program 25 judges that the judgment as to whether the character string is in an upright state or in an inverted state cannot be performed, for the reason that all the English characters that constitute the character string are capital letters, or for other reasons. In this case, the orientation judgment program 25 proceeds to step S3~~1~~8. In step S3~~1~~8, the orientation judgment program 25 performs processing for starting the second orientation judgment program 26.